Collaborative Autonomy with Group Autonomy for Mobile Systems (GAMS)

Presenter: James Edmondson
(jredmondson@sei.cmu.edu)

Date: November 18, 2014

Software Engineering Institute | Carnegie Mellon

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
18 NOV 2014	**N/A**	

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Collaborative Autonomy with Group Autonomy for Mobile Systems (GAMS)	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
Edmondson /James R.	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited.

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

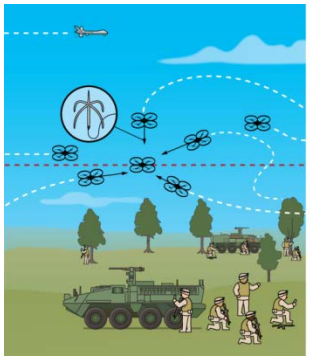
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**SAR**	**20**	

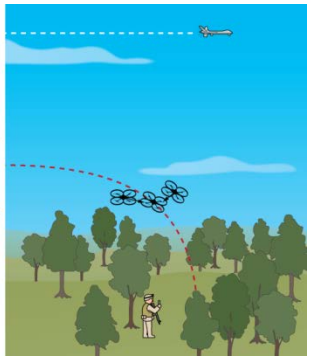
This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

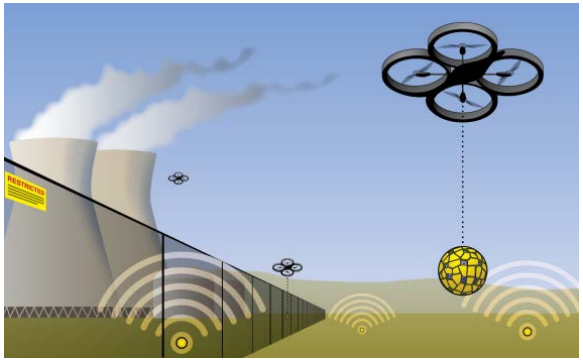
DM-0001800

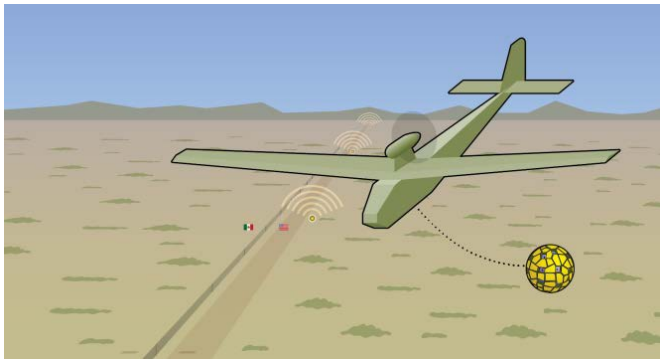
Problems facing group autonomy

- Autonomy focus is on single unit control
- Focus is on centralized controllers (prone to failure/attack)
- Autonomy frameworks tend to be targeted at homogeneous platforms and algorithms
- Blocking communications are prone to faults/attacks/outages/loss-of-control
- GPS is highly inaccurate for precise maneuvers
- Lack of standardization for autonomous collaboration



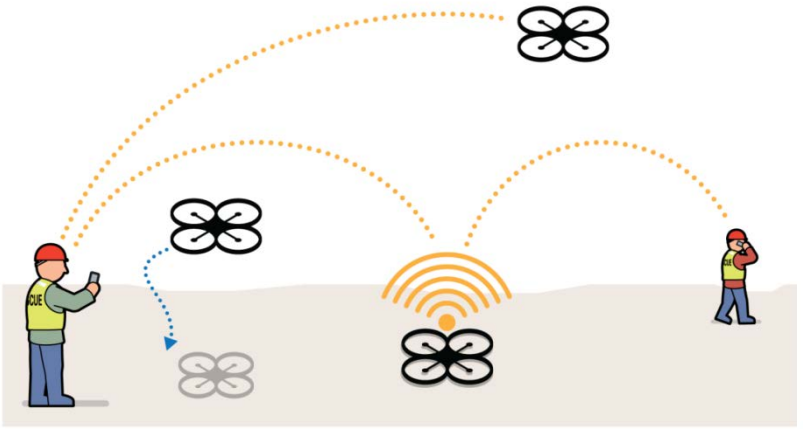






Our Approach to Group Autonomy

1. Create a portable, open-sourced, decentralized operating environment for autonomous control and feedback. Focus on scalability, performance and extensibility
2. Integrate the operating environment into unmanned autonomous systems (UAS), platforms, smartphones, tablets, and other devices. Focus on portability.
3. Design algorithms and tools to perform mission-oriented tasks like area coverage and network bridging between squads
4. Design user interfaces to help single human operators control and understand a swarm of UAS, devices, and sensors (human-on-the-loop autonomy)

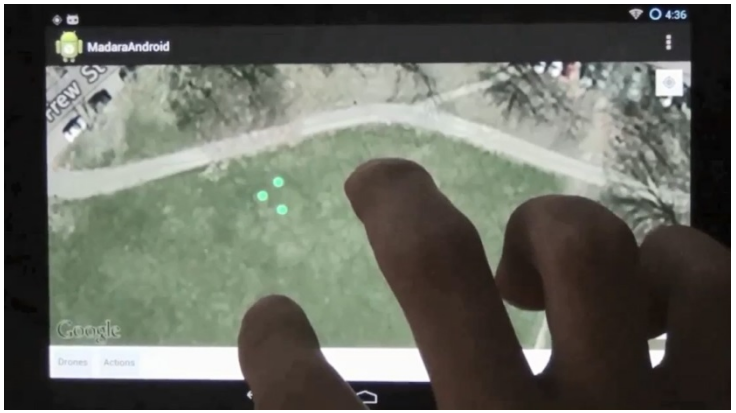


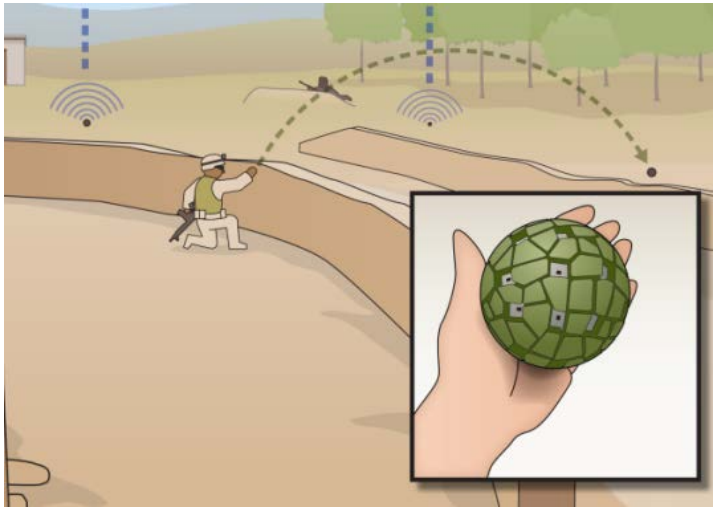
FY 2014 Technologies/Platforms

We investigated several platforms and collaborations in FY 2014, including:

- **UAVs (Parrot and 3D Robotics)**
- **Throwables (Bounce Imaging), Smartphones, Tablets (Android)**
- **High precision and gps-denied positioning**

FY 2015 is focusing on autonomous swarms of 25+ boats (Platypus/CMU collaboration)



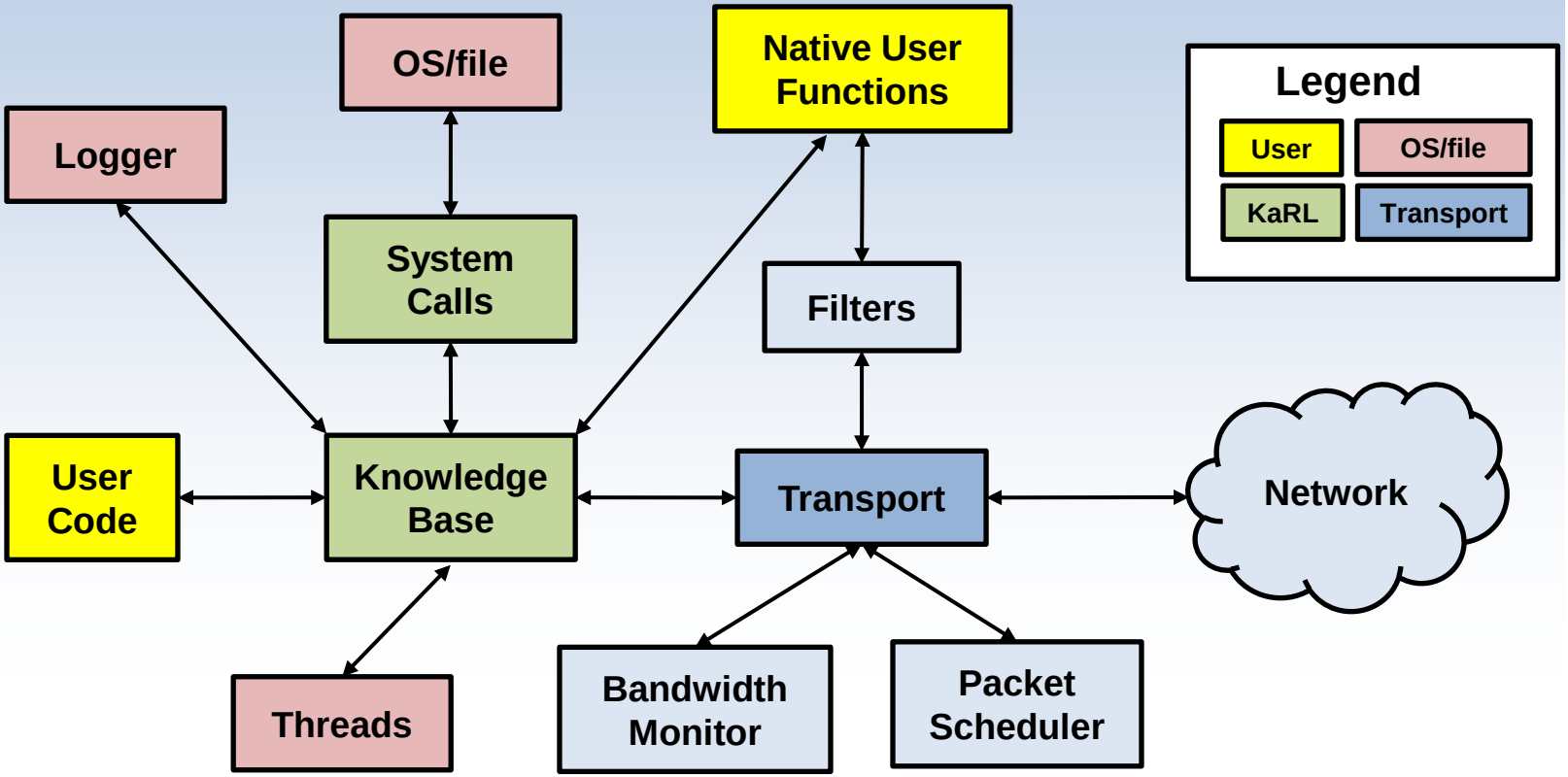


Principles of our open-sourced middleware (MADARA and GAMS)

1. Be useful to application developers
2. Enable distributed, decentralized artificial intelligence
3. Be fast, small, and capable
4. Be portable to as many platforms relevant to UAS as possible
5. Be extensible to facilitate new transports, linking with external libraries, security, assurance, and consistency
6. Provide extensive documentation

MADARA Architecture

More information, tutorials, and documentation at http://madara.googlecode.com



Key MADARA Features (2009-present)

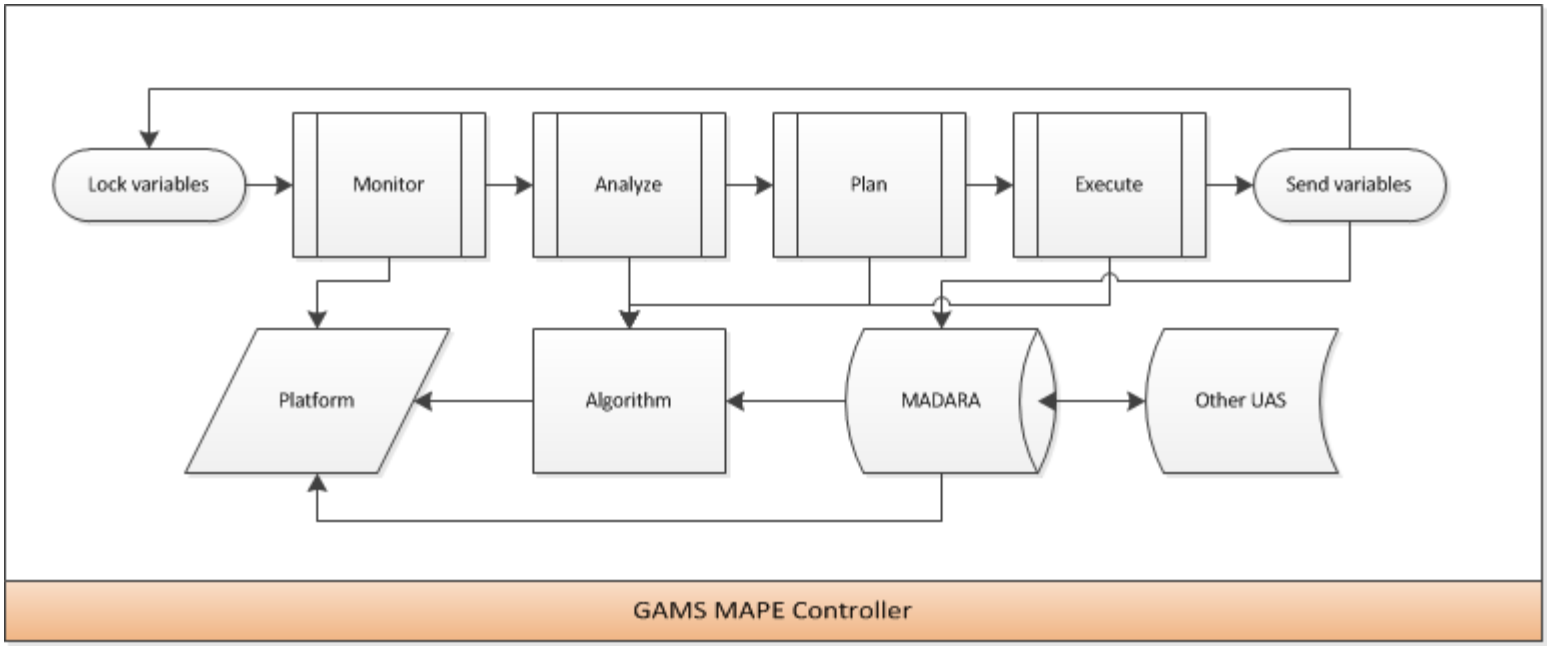
- Allows developers to write both state-based and event-based programs (or combinations of both) for distributed artificial intelligence
  - Programs can react to receive, send, or rebroadcast events
  - Programs can have deadline-enforced periodic executions, wait for certain state-based conditions to come true, or execute efficient, dynamic actions in KaRL (Knowledge and Reasoning Language)
- Provides object-oriented containers and threads as first class entities
- Supports C++, Java, Python, ARM, Intel, Windows, Linux, Android, iOS
- Supports IP multicast, broadcast, unicast, OMG DDS transports
- Enforces consistency of updates through Lamport clocks, priorities
- Extensible transport layer, filtering system, and callbacks
- Extensive documentation (guides, tutorials, doxygen)

How MADARA helps researchers and developers

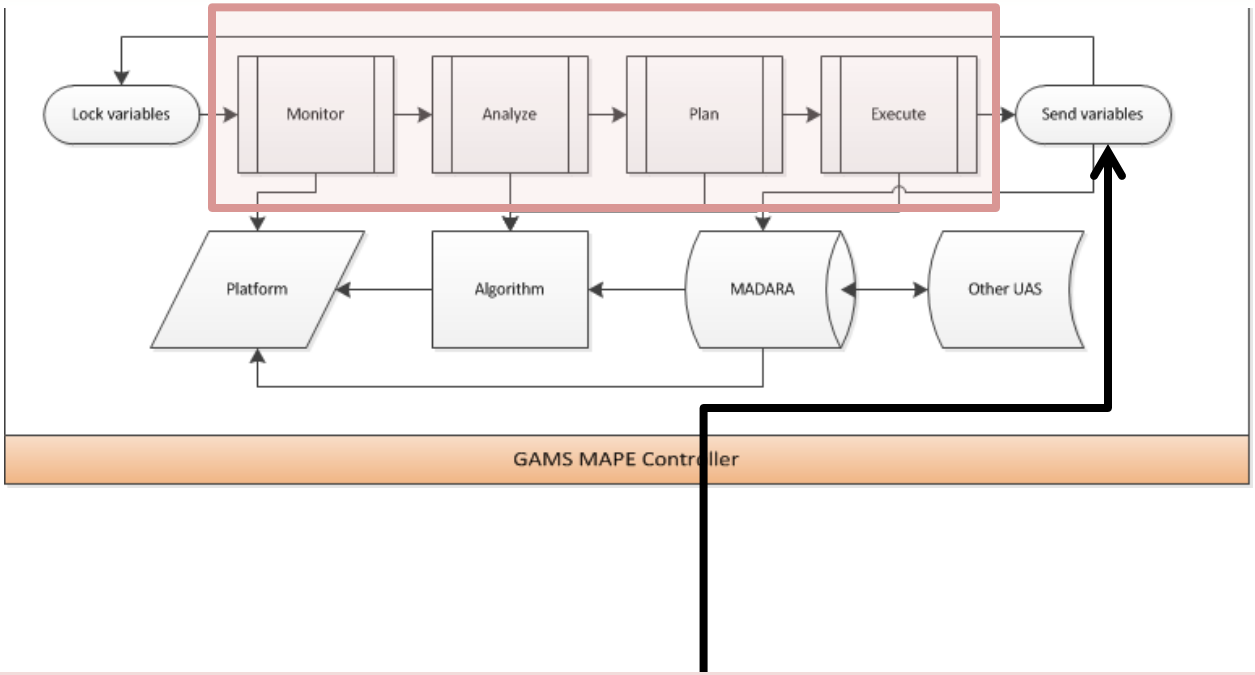
- **Facilitates distributed and multithreaded programming**
  - Networking and threading is provided
  - Performance and scaling is exceptional
  - Language and architecture portability to prevent vendor lock-in and shorten transition timeframe
  - Open source. Free. Extensible.
- **Allows reseachers to focus on what is important to them**
  - Quickly code and experiment with multi-processed, multi-threaded, or multi-robot applications with dependable, portable code
  - Scale to thousands of collaborating entities in real-time

GAMS Architecture (FY 2014)

1. Built directly on top of MADARA
2. Utilizes MAPE loop (IBM autonomy construct)
3. Provides extensible platform, sensor, and algorithm support
4. Uses new MADARA feature called containers, which support object-oriented programming of the Knowledge Base



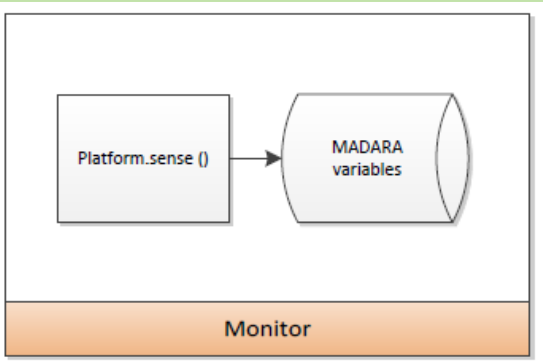
GAMS Architecture (FY 2014)

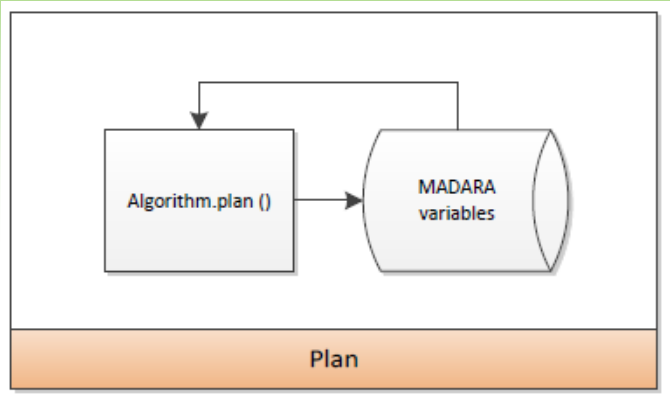


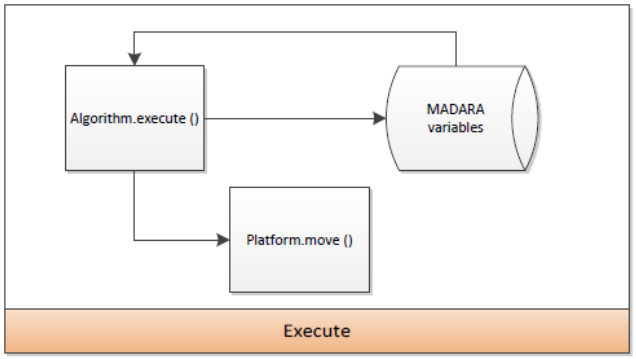
Key points:

- **During the MAPE loop, the context is locked from external updates**
- **At the end of the MAPE loop, all global variable changes are aggregated together and sent to other UAS participating in the mission**

GAMS Platform and Algorithm Interactions

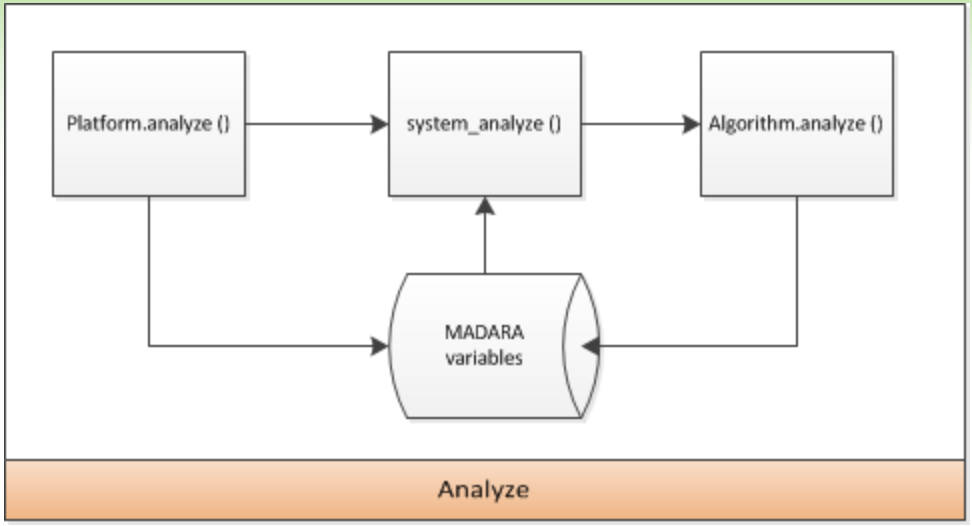






The Monitor, Plan, and Execute phases are pretty straight-forward

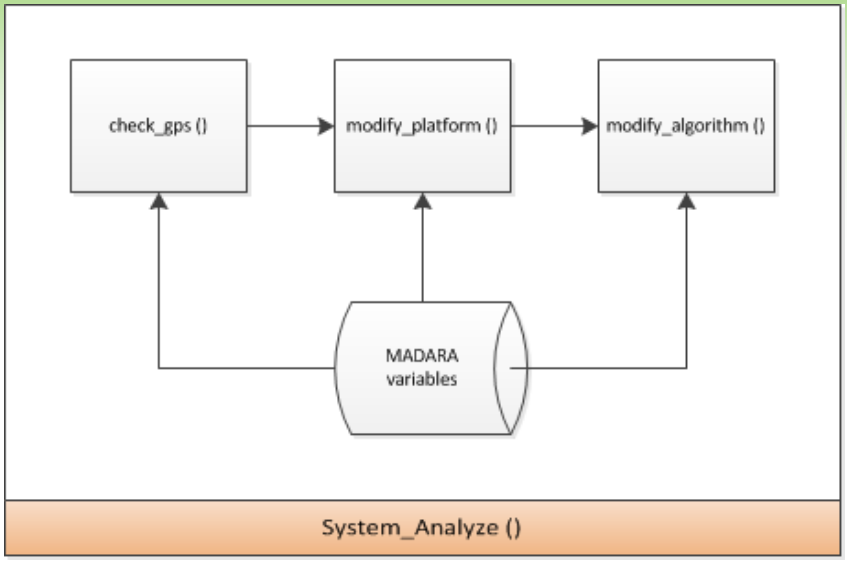
GAMS Platform and Algorithm Interactions



During the analyze phase:

1. **The platform analyzes its state and informs the rest of the GAMS system via MADARA variables**
2. **The system analyzes the platform and environment for algorithm changes**
3. **The algorithm then analyzes its state and sets appropriate MADARA variables.**

GAMS Platform and Algorithm Interactions



About system_analyze ():

1. **The platform can inform the control loop of gps-spoofing, if it has capabilities**
2. **Check_gps () is also intended to implement gps-spoof checking in software**
3. **Environmental or platform characteristics can result in changes to the platform (e.g., an arm is damaged) or algorithm (e.g., the UAS should return home)**

How to use GAMS with new platforms and algorithms

1. Extend the platform base class
   - Implement move (), land (), takeoff (), or other functions
   - Implement sense ()
   - Implement analyze ()
2. Extend the algorithm base class
   - Implement analyze ()
   - Implement plan ()
   - Implement execute ()
3. Extend the base controller class (optional)
   - Override MAPE methods
4. Use the parameterized Mape_Loop class (optional)
   - Use the define_monitor, define_analyze, etc. methods with MADARA functions

What exactly are we solving?

1. MADARA is a bit expansive in its capabilities and developers can find themselves pulled in many different directions when thinking of autonomy to implement. **GAMS provides an interface for algorithms and platforms to be added and utilized within a wireless environment**
2. **GAMS provides mechanisms for tracking platform and algorithm states and characteristics**, such as detection of GPS-spoofing, blocked/deadlocked conditions within algorithms, low battery, degraded sensors, etc.
3. While MADARA may support any type of distributed artificial intelligence paradigm, **GAMS provides a stable, consistent framework for group autonomous behaviors and may prove beneficial to standardization efforts for group autonomy**

New Features in FY 2015

1. **Tighter and more feature rich MADARA interactions**
   - GAMS may now be directly ran inside of MADARA threads (October 2014)
   - GAMS may now run at multiple hertz speeds for sampling sensors at varying rates
   - GAMS may have separate sampling and sending hertz rates
2. **Multiple platform support in VREP and real-world**
   - VREP: Quadcopter, Ant, and possibly Boat models and platforms
   - Real World: Drone-RK quadcopter and Platypus boat
3. **Even more focus on scale and reliability**

FY 2015 Goals and Objectives (ELASTIC Project)

1. Showcase GAMS and MADARA on 25+ real, collaborating robots
   - Focusing on Paul Scerri's robotic boats (Platypus/CMU)
2. Facilitate transition of GAMS/MADARA into DARPA or DoD Labs
3. Quantify scalability limitations
4. Identify best practices for developing distributed mission-focused autonomy applications

Carnegie Mellon University

Software Engineering Institute
Carnegie Mellon

Closing remarks

In this talk, we've discussed

- A **distributed reasoning engine called MADARA** that provides portable, fast reasoning services for distributed artificial intelligence
- An **extensible framework called GAMS for distributed algorithms and platforms** that enables Monitor-Analyze-Plan-Execute-based distributed autonomous systems

Carnegie Mellon University

Software Engineering Institute
Carnegie Mellon

FY 2014 Open Source Release

The algorithms, tools, and middleware created at SEI are released via BSD-style licenses through the following projects:

- Multi-Agent Distributed Adaptive Resource Allocation (MADARA) for the distributed OS layer: http://madara.googlecode.com
- Group Autonomy for Mobile Systems (GAMS) for the algorithms and UIs: http://gams-cmu.googlecode.com
- Model Checking for Distributed Applications (MCDA) http://mcda.googlecode.com
- Drone-RK for the UAV device drivers: http://www.drone-rk.org
- Contact: jredmondson@sei.cmu.edu

SEI Project Members

James Edmondson

Sagar Chaki

Sebastian Echeverria

Gene Cahill

CMU Project Members

Anthony Rowe

Oliver Shih

Luis Pinto

Vanderbilt Students

Anton Dukeman (CS)

Subhav Pradhan (ISIS)

Carnegie Mellon University

Software Engineering Institute
Carnegie Mellon